

FIG. 1

```
14 {
    * 1. productive class :
    * defination
    class OPERATIONS definition.
    public section.
    class-methods:
    14a {
        ADD importing A type I
        B type I
        returning VALUE (RESULT) type I.
    }
    endclass.

    * implementation
    class OPERATIONS implementation.
    14b {
        method ADD.
        RESULT = A + B.
        endmethod.
    }
    endclass.
}

16 {
    * 2. test class:
    * definition
    18a {
        class TEST_OPERATIONS definition for testing.
        public section.
        methods TEST_ADD for testing.
    }
    endclass.

    * implementation
    class TEST_OPERATIONS implementation.

    method TEST_ADD.

    * test data: variable needed to store the result from the productive method
    data: ACTUAL_RESULT type I.

    * call the method under test:
    ACTUAL_RESULT = OPERATIONS=>ADD ( A = 3 B = 5 ).

    18b {
        * compare the result with the expected value:
        CL_AUNIT_ASSERT=>ASSERT_EQUALS (
            ACT = ACTUAL_RESULT
            EXP = 8
            MSG = 'this is the message which occurs if the test failed'
        ).
    }

    endmethod.
    endclass.
}
```

FIG. 2

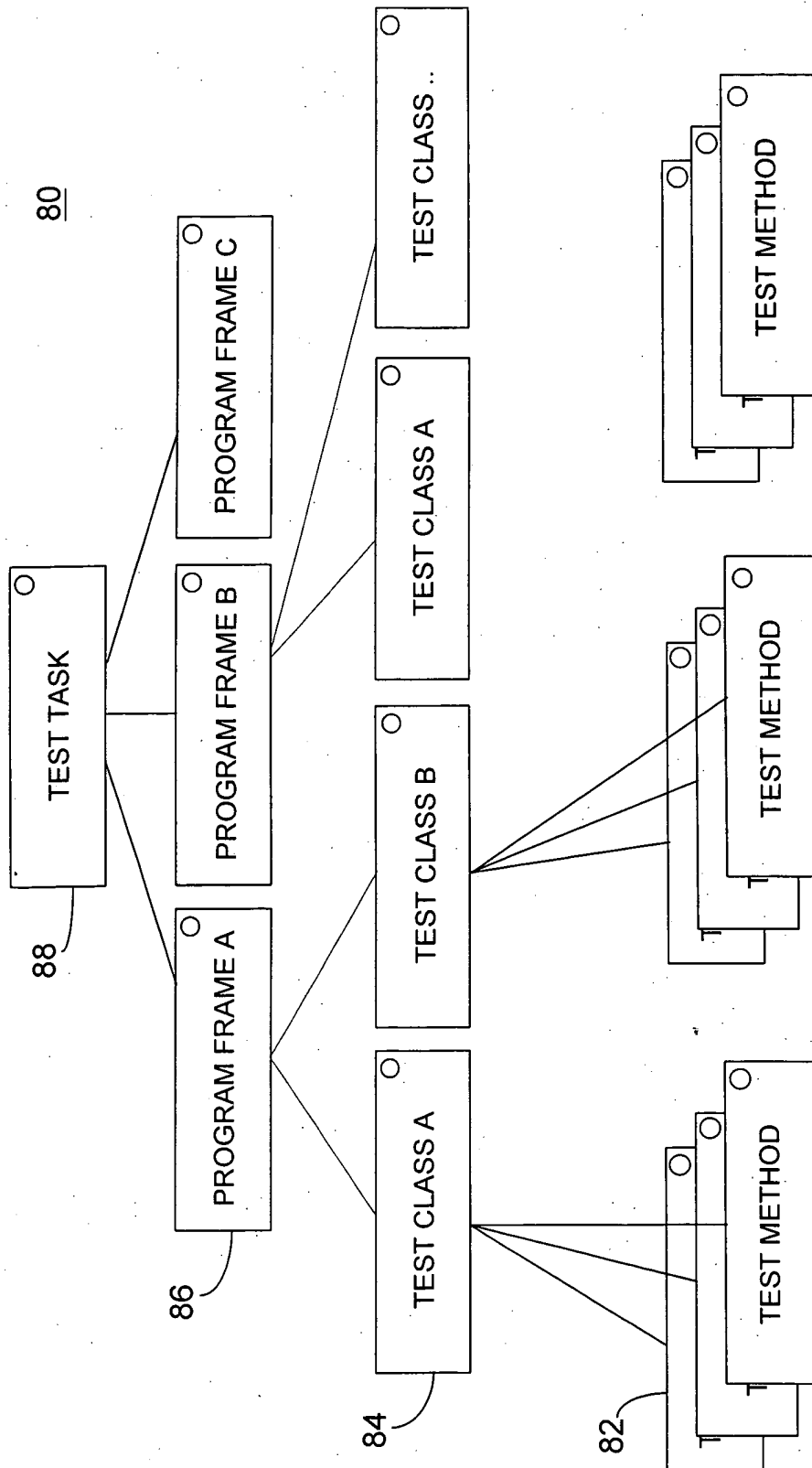


FIG. 3

56
ASSERT_EQUALS (ACT = ACTUAL RESULT

EXP = EXPECTED_RESULT
MSG = 'this test has failed'
QUIT = QUIT_VALUE).
57 58

Where QUIT_VALUE defines at which level the test flow should be interrupted:

- NO: continue the current test method.
- METHOD: interrupt the current test method.
- CLASS: interrupt the test class execution.
- PROGRAM: abandon all test class executions of the currently tested program frame.

FIG. 4

